---
title: "Hierarchical Models in Practice Exercises"
date: "September 17, 2017"
output: html_document

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```
```{r}
# Load the following data (you can install it from Bioconductor) and extract the data matrix using exprs:

## to install:
library(rafalib)
install_bioc("SpikeInSubset")
library(Biobase)
library(SpikeInSubset)
data(rma95)
y <- exprs(rma95)

# If you have trouble installing the  SpikeInSubset package please consult the discussion boards.
#
# This dataset comes from an experiment in which RNA was obtained from the same background pool to create six replicate samples. Then RNA from 16 genes were artificially added in different quantities to each sample. These quantities (in picoMolars) and gene IDs are stored here:

pData(rma95)

#Note that these quantities were the same in the first three arrays and in the last three arrays.
#So we define two groups like this:

g <- factor(rep(0:1,each=3))

#and create an index of which rows are associated with the artificially added genes:

spike <- rownames(y) %in% colnames(pData(rma95))
```

1. Note that only these 16 genes are differentially expressed since these six samples differ only due to random sampling (they all come from the same background pool of RNA).

Perform a t-test on each gene using the rowttests function in the genefilter package.

What proportion of genes with a p-value < 0.01 (no multiple comparison correction) are not part of the artificially added (false positive)?
unanswered

```{r}
source("http://bioconductor.org/biocLite.R")
biocLite()
biocLite("genefilter")

library(genefilter)

tt = rowttests(y,g)
index = tt$p.value < 0.01
( mean( !spike[index] ))
```
```{r}
## We can make a volcano plot to visualize this:
rtt = rowttests(y,g)
mask <- with(rtt, abs(dm) < .2 & p.value < .01)
cols <- ifelse(mask,"red",ifelse(spike,"dodgerblue","black"))
with(rtt,plot(-dm, -log10(p.value), cex=.8, pch=16,
     xlim=c(-1,1), ylim=c(0,5),
     xlab="difference in means",
     col=cols))
abline(h=2,v=c(-.2,.2), lty=2)
```
2. Now compute the within group sample standard deviation for each gene (you can use group 1). Based on the p-value < 0.01 cut-off, split the genes into true positives, false positives, true negatives and false negatives. Create a boxplot comparing the sample SDs for each group. Which of the following best described the box-plot?

    The standard deviation is similar across all groups.
    On average, the true negatives have much larger variability.
    The false negatives have larger variability.
    The false positives have smaller standard deviation. correct

```{r}
wg1_sds <- rowSds(y[,g==0])
index <- paste0( as.numeric(spike), as.numeric(tt$p.value<0.01))
index <- factor(index,levels=c("11","01","00","10"),labels=c("TP","FP","TN","FN"))
boxplot(split(wg1_sds,index))

```

3. In the previous two questions we observed results consistent with the fact that the random variability associated with the sample standard deviation leads to t-statistics that are large by chance.

Note that the sample standard deviation we use in the t-test is an estimate and that with just a pair of triplicate samples, the variability associated with the denominator in the t-test can be large.

The following three steps perform the basic limma analysis. The eBayes step uses a hierarchical model that provides a new estimate of the gene specific standard error.

```{r}
library(limma)
fit <- lmFit(y, design=model.matrix(~ g))
colnames(coef(fit))
fit <- eBayes(fit)

#Make a plot of the original new hierarchical models based estimate versus the sample based estimate.

sampleSD = fit$sigma
posteriorSD = sqrt(fit$s2.post)

LIM = range( c(posteriorSD,sampleSD))
plot(sampleSD, posteriorSD,ylim=LIM,xlim=LIM)
abline(0,1)
abline(v=sqrt(fit$s2.prior))

```

Which best describes what the hierarchical modelling approach does?

- Moves all the estimates of standard deviation closer to 0.12. correct
- Increases the estimates of standard deviation to increase t.
- Decreases the estimate of standard deviation.
- Decreases the effect size estimates.

4. Use these new estimates (computed in Question 4.6.3) of standard deviation in the denominator of the t-test and compute p-values. You can do it like this:

```{r}
library(limma)
fit = lmFit(y, design=model.matrix(~ g))
fit = eBayes(fit)
##second coefficient relates to diffences between group
pvals = fit$p.value[,2]
```

What proportion of genes with a p-value < 0.01 (no multiple comparison correction) are not part of the artificially added (false positives)?

```{r}
pvals <- fit$p.value[,2]
new_index <- pvals < 0.01
( mean( !spike[new_index] ) )
```

```{r}
## We can make a volcano plot to visualize this:
mask <- abs(fit$coef[,2]) < .2 & fit$p.value[,2] < .01
cols <- ifelse(mask,"red",ifelse(spike,"dodgerblue","black"))
plot(fit$coef[,2], -log10(fit$p.value[,2]), cex=.8, pch=16,
     xlim=c(-1,1), ylim=c(0,5),
     xlab="difference in means",
     col=cols)
abline(h=2,v=c(-.2,.2), lty=2)
```

Compare to the previous volcano plot and notice that we no longer have small p-values for genes with small effect sizes.